PocketMenu: Non-Visual Menus for Touch Screen Devices

Martin Pielot, Tobias Hesselmann, Wilko Heuten
OFFIS Institute for Information Technology
Oldenburg, Germany
pielot;hesselmann;heuten@offis.de

Anastasia Kazakova, Susanne Boll
University of Oldenburg
Oldenburg, Germany
anastasia.kazakova@uni-oldenburg.de
boll@informatik.uni-oldenburg.de

ABSTRACT
We present PocketMenu, a menu optimized for non-visual, in-pocket interaction with menus on handheld devices with touch screens. By laying out menu items along the border of the touch screen its tactile features guide the interaction. Additional vibro-tactile feedback and speech allows identifying the individual menu items non-visually. In an experiment, we compared PocketMenu with iPhone's VoiceOver. Participants had to control an MP3 player while walking down a road with the device in the pocket. The results provide evidence that in the above context the PocketMenu outperforms VoiceOver in terms of completion time, selection errors, and subjective usability. Hence, it enables usage of touch screen apps in mobile contexts (e.g. walking, hiking, or skiing) and limited interaction spaces (e.g. device resting in a pocket).

ACM Classification Keywords
H.5.2 User Interfaces: Haptic I/O

Author Keywords
Handheld Devices and Mobile Computing; Input and Interaction Technologies; Tactile & Haptic UIs

INTRODUCTION
Regarding mobile devices, there is a clear trend towards removing physical buttons in favor of a large touch screen. The best example is the iPhone, which offers a single "home" key on its front while all the other interaction is done via the touch screen. The advantage over hardware interfaces is that the interactive elements on touch screens allow to better adapt the interface to each application.

However, interacting with touch screens places heavy visual load on the user. Even if the user learns the layout of the user interface – which often happens with physical interfaces – it still would be close to impossible to hit the correct item blindly. This problem increases when additionally the user is on the move (e.g. walking, cycling, or skying) or using the device in constraint spaces, such as the jacket pocket.

While many approaches existing to make touch screens accessible for non-visual use [1, 2, 3, 4, 5, 6, 10, 11], they might not cope well with the limited interaction space and the not-so-well-known orientation of the device when the device rests in a pocket. The PocketMenu[1] exploits the fact that most handheld's screen borders have tactile features and thus can be felt. Menu items are laid out along the border so it provides a means of orientation. By adding vibro-tactile feedback when crossing the border of an item, it enables users to navigate a menu purely through touch.

MobileHCI'12, September 21–24, 2012, San Francisco, CA, USA.

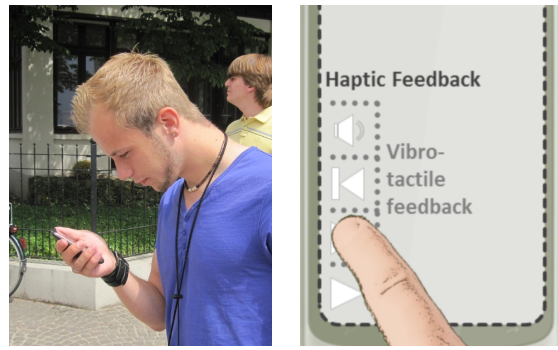

Figure 1. To avoid having to look at touch screen for interacting with them (left) we propose the PocketMenu (right).

In a field experiment we compared the PocketMenu against iPhone's VoiceOver. The results suggest that the concept of the PocketMenu has significant positive advantages when used eyes-free and in the pocket.

RELATED WORK
Several novel input techniques have been proposed to allow eyes-free interaction with touch screens. For example, PieMenus allow users to spawn a menu at any point on the touch screen. Its items appear around the touch points. Items are selected by swiping towards them. Combined with audio feedback, PieMenus even allow blind people to enter text on a touch screen [1]. SlideRule [6] combines gestures with audio feedback to enable blind users to interact with touch screens. Compared to a button-based system it was faster but more error prone. Recently, Wilson et al. [9] proposed pressure-based menus and showed that these are well suited for eyes-free interaction on the move.

Another approach to improve the accessibility of touch screens is to use haptic feedback. EdgeWrite [10] uses a plastic template with a small hole to limit the movement of a stylus used for gesture-based character input. Wobbrock et al.

[1] **https://play.google.com/store/apps/details?id=org.pielot.pocketmp3**

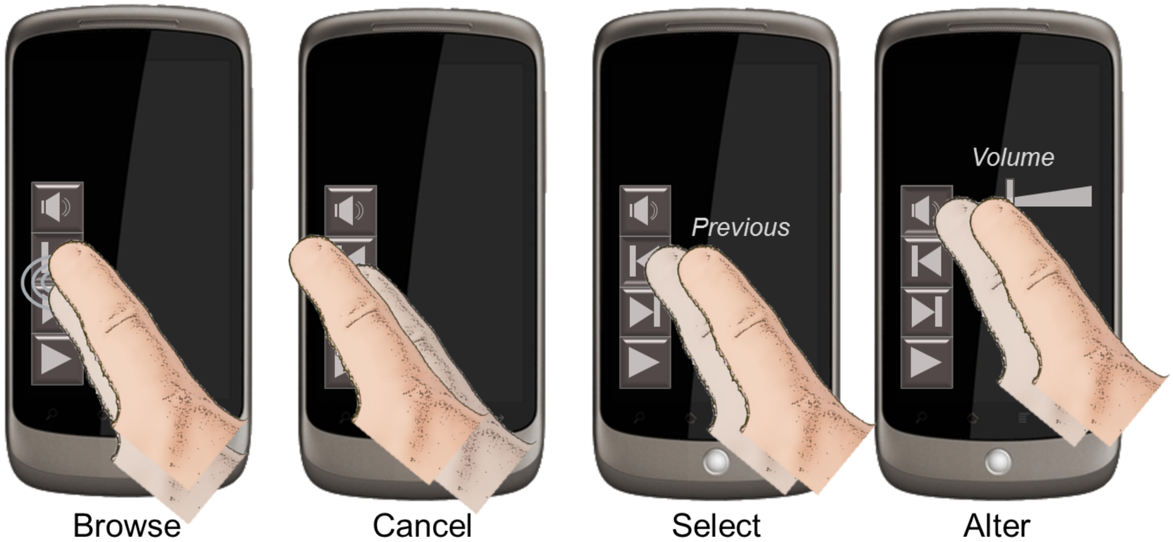

Figure 2. The four main functions of the PocketMenu: browse, cancel, select, and adjust.

show that text entry with EdgeWrite is faster than free letter recognition. Barrier Pointing [4] aims at making stylus-based touch screen interaction more accessible for motor impaired users by utilizing the raised borders of the screen that can be found in some devices. The menu is arranged along the border of the screen. The raised border is used to restrict and guide the stylus input. McGookin et al. [8] used raised paper to make touch screens accessible for the blind. These previous world show that haptic feedback can effectively increase the accessibility of touch screens.

A related approach is using vibro-tactile feedback. T-Bars [5] introduce stroke-formed menu items which trigger vibration when being touched. To select an item the user slides along the stroke until its end. Pilot evaluations showed that this feedback helps users to keep their fingers on the line more accurately. Rich tactile feedback is used by SemFeel [11] to not only indicate the presence of an item on the touch screen but also to convey the item's type.

DESIGN

While previous work shows how to facilitate non-visual access to touch screens, PocketMenu specifically aims at supporting interaction on the move with the device in the pocket. These constraints make it difficult to directly apply previous approaches. The unknown orientation of the device impedes hitting the desired element on the screen. Restricted space for physical interactions hinders performing gestures.

Layout. As a solution, PocketMenu (see Fig. 2) combines previous approaches into a new concept. The menu items are laid out along the screen's border. Similar to EdgeWrite [10], the screen border serves as tactile reference, guiding the user's finger when browsing the menu's items. Vibro-tactile feedback, as e.g. used in T-Bars [5], is used to indicate the border between menu items. The lower-left edge of the touch screen serves as the menu's origin, as edges are physical forms that are identified easily by the sense of touch [7].

Browse Items. To browse the menu, the user slides her/his finger along the screen's border. A short tactile pulse of 100 ms indicates that the finger crosses the border between two items. Menu items can be identified in two ways. First, when the finger enters a menu item the associated action is announced via text-to-speech. Second, when the users start building a mental model of the menu, they can just remember the items' position and count the number of vibro-tactile pulses when sliding along the screen border, which allows to use the menu without speech output if desired.

Cancel Browsing. Browsing the menu can be aborted by simply releasing the finger. In the related work, e.g. SemFeel [11], releasing the finger has been used to trigger the last selected item. However, due to the constraints we were afraid that users too often might lift the finger by accident. Our release-to-cancel approach avoids these accidents.

Select and Alter. The PocketMenu offers two kinds of items: button-like items that trigger associated actions (e.g. Play or Pause) and slider-like items that allow to continuously adjust a value (e.g. Volume). The selection of button items is inspired by the classic PieMenu [2]: by swiping the finger towards the middle of the screen the associated action is executed. A short tactile pulse ($100ms$) confirms that the button item has been 'pressed'. Slider items are selected in a similar way. The associated value can be adjusted by swiping more or less far away from the menu border before lifting the finger. A series of short tactile pulses ($< 10ms$) issued when moving the finger indicate that the associated value is altered.

USER STUDY

To study how well PocketMenu enables in-pocket-use on the move, we conducted a field experiment, where we compared it with a replica of VoiceOver available on iPhone 3GS and later models.

Methodology

Material. We implemented a simple MP3 player as test application. It offers five different functions: start/pause, next, prev, volume up, volume down. As baseline for the study we replicated the relevant interaction concepts of the iPhone's *VoiceOver* functionality and implemented menu that resembled a typical MP3 player control (see Fig. 3). Using the VoiceOver clone users can explore the screen elements by moving the finger over the screen elements. When hitting an element a short click sound is played the associated action function is announced via speech. To execute the action of the recently touched element the user has to perform a double-tap anywhere on the screen. To alter adjustable elements, such as the volume slider, the finger needs to remain on the screen after the second tap hit the screen. Then the element can be continuously be altered by swiping left or right. Alternatively, the user can double-tap onto the slide to jump the cursor to the tapped position.

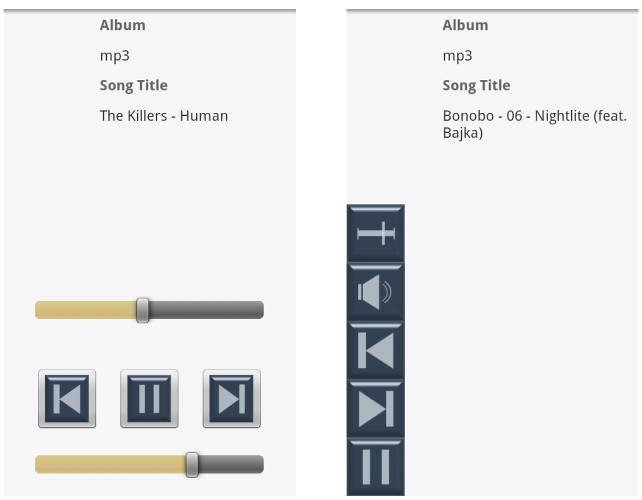

Figure 3. Two screenshots of the evaluated menus. On the left the layout of the *VoiceOver* enhanced menu is shown (control condition), The right screenshot shows the layout of the *PocketMenu* (experimental condition).

Participants. Ten participants (4 female) with an average age of 25.4 (SD 2.95) took part in the study. The average experience with touch screens was rated above average with a considerable spread (3.40, SD 1.58, on a five-point Likert-scale, 1=low, 5=high).

Design. The menu type served as independent variable with two levels: VoiceOver, PocketMenu. We used a within-subject design with a counter-balanced order of conditions. As performance measures, the software automatically recorded the *completion time* (the time it takes the participants to execute a given command) and the *selection errors* (number of times the participants execute the wrong action). Furthermore, we noted the number of *breakdowns* (participant has to take device out of the pocket to execute command) and we approximated the cognitive workload by measuring the *effective walking speed*. The standardized SUS questionnaire was used to quantify the menus' *subjective usability*.

Procedure. At the beginning of each session, the participants could familiarize themselves with both menus. This typically took 10 to 15 minutes. Afterwards, the evaluation was continued outside. The participants were asked to walk down a pedestrian a sidewalk with the phone in the pocket of their jacket. The software announced the action to be executed (e.g. "press play" or "change the volume"). The participants repeated the announcement, so the experimenter had the chance to correct any misunderstanding. Then, the participants tried to execute the action as fast as possible while avoiding mistakes. 13 items had to be selected with each menu. The interaction with the menu (*completion time* & *selection errors*) was logged automatically by the application. The *breakdowns* and the walked distance, which was used to infer the *effective walking speed*, were noted by the experimenter. Afterwards, the participants were asked to fill out two SUS questionnaires, one for each menu. Finally, a short open interview was conducted to learn about the participants' impressions and thoughts on both designs.

Results

Table 1 summarizes mean and standard deviation for each dependent measure.

Table 1. Quantitative Results

	VoiceOver	*PocketMenu*
Completion Time (s) ***	18.13 (22.33)	11.12 (12.28)
Selection Errors (#) **	1.68 (5.84)	0.51 (1.67)
Breakdowns (#)	1.80 (1.81)	1.00 (1.70)
Walk. Speed (km/h)	3.21 (2.06)	3.44 (1.47)
Usability (SUS) **	50 (28.6)	78.5 (19.7)

PocketMenu users selected items significantly faster (*completion time*) ($p < .001$), made significantly fewer *selection errors* ($p < .01$), and provided significantly higher *subjective usability* ratings ($p < .001$). No significant effects could be found on the number of *breakdowns* between the conditions ($p = .10$) and the *effective walking speed* ($p = .27$).

The participants appreciated that for the PocketMenu the screen's border provided orientation. Not knowing about the tests aims they proposed to rotate the VoiceOver menu's content by 90 so the two sliders would be next to the screen's border to simplify the interaction with them.

The PocketMenu's swipe technique was preferred over the VoiceOver's double-click to execute menu items. The main point of criticism about the VoiceOver menu was that it was found difficult to perform double-clicks in the pocket. However, it was also appreciated that the double-click selection made it easier to perform repeated selection of the same item.

When browsing the PocketMenu, many participants appreciated the tactile feedback. The speech output, in contrast, was often considered annoying after repeated usage. VoiceOver also added an auditory click whenever a button was touched, which was found to be adding to the "cacophony of sounds". Two participants reported to have browsed the PocketMenu based on the tactile feedback only and ignored the speech announcements altogether. Suggestions were to allow the user to turn off the speech output or replace it by more subtle non-speech audio or tactile feedback.

Discussions

In summary, the PocketMenu allowed the participants to execute actions in less time and with fewer errors. The participants appreciated the haptic feedback and rated its usability higher.

There are three design difference between the PocketMenu and the VoiceOver menu, which are potential causes for these observed effects: (1) in the PocketMenu the screen border guides the interaction by providing tactile feedback, (2) buttons provide speech + vibro-tactile instead of speech + auditory feedback, and (3) elements are executed by a swipe gesture instead of a double-click.

From the participants' feedback we learned that the haptic feedback provided by the border, as suggested by Wobbrock et al. [10] for stylus-based touch screen interaction can successfully be transferred to finger-based interaction. According to the qualitative feedback, this was the key design element to support usage on the move.

Our study's results confirms that vibro-tactile feedback, as proposed by Hall et al. [5] or by Yatani and Truong [11], is a highly efficient feedback mechanism. Unlike the auditory feedback, it did not become annoying. For the PocketMenu, some participants suggested to allow turning of the speech output completely, since they felt able to find the right button via vibration feedback only. Other participants suggested replacing speech announcements through vibration patterns, which has been found useful in previous work [11]. We conclude that tactile feedback has the potential to replace annoying audio feedback. On the basis of these findings, we informally tested replacing the VoiceOver's click with a vibrotactile buzz and believe to have observed a notable performance boost.

With respect to the executing of menu items our results support the guideline established by McGookin et al. [8] to avoid double clicks in non-visual touch screen interaction. Further, the results support the findings by Ecker et al. [3] and Bonner et al. [1] that swiping is an effective and efficient gesture for interacting non-visual with touch screens.

CONCLUSIONS

With the PocketMenu, we introduced a menu design that enables eyes-free, in-pocket use of handheld touch screen devices on the move. It guides the interaction by haptic feedback from the raised border of the screen and vibration pulses created by the internal pager motor. In a field study with 10 participants, who had to control an MP3 player on the move and with the device in their pocket, we experimentally compared the PocketMenu with a replica of iPhone's VoiceOver solution. The results provide evidence that the added tactile feedback allows faster interaction with fewer errors and improves the subjective usability.

The PocketMenu enables designing touch screen applications that users can controls without dedicating visual attention to it. This applies to all applications that convey their information in non-visual formats and that are used in mobile situations where the visual attention is desired/required elsewhere, such as walking, driving, cycling, or skiing. Since many countries discuss legislative approaches to discourage mobile phone use in traffic due to visual distraction, enabling eyes-free usage of touch screen UIs may even become mandatory in some contexts, such as information systems in cars.

ACKNOWLEDGMENTS

The authors are grateful to the European Commission which co-funds the IP HaptiMap (FP7-ICT-224675).

REFERENCES

1. Bonner, M. N., Brudvik, J. T., Abowd, G. D., and Edwards, W. K. No-look notes: Accessible eyes-free multi-touch text entry. In *Pervasive* (2010).
2. Callahan, J., Hopkins, D., Weiser, M., and Shneiderman, B. An empirical comparison of pie vs. linear menus. In *Proc. CHI* (1988).
3. Ecker, R., Broy, V., Butz, A., and De Luca, A. pietouch: a direct touch gesture interface for interacting with in-vehicle information systems. In *Proc. MobileHCI* (2009).
4. Froehlich, J., Wobbrock, J. O., and Kane, S. K. Barrier pointing: using physical edges to assist target acquisition on mobile device touch screens. In *Proc. ASSETS* (2007).
5. Hall, M., Hoggan, E., and Brewster, S. T-bars: towards tactile user interfaces for mobile touchscreens. In *Proc. MobileHCI* (2008).
6. Kane, S. K., Bigham, J. P., and Wobbrock, J. O. Slide rule: making mobile touch screens accessible to blind people using multi-touch interaction techniques. In *Proc. ASSETS* (2008).
7. Lederman, S. J., and Klatzky, R. L. Haptic identification of common objects: effects of constraining the manual exploration process. *Percept Psychophys. 66*, 4 (2004), 618–628.
8. McGookin, D., Brewster, S., and Jiang, W. Investigating touchscreen accessibility for people with visual impairments. In *Proc. NordiCHI* (2008).
9. Wilson, G., Brewster, S. A., Halvey, M., Crossan, A., and Stewart, C. The effects of walking, feedback and control method on pressure-based interaction. In *Proc. MobileHCI* (2011).
10. Wobbrock, J. O., Myers, B. A., and Kembel, J. A. Edgewrite: a stylus-based text entry method designed for high accuracy and stability of motion. In *Proc. UIST* (2003).
11. Yatani, K., and Truong, K. N. Semfeel: a user interface with semantic tactile feedback for mobile touch-screen devices. In *Proc. UIST* (2009).